Views and Documenting Software Architecture

March 2014

Ying SHEN

SSE, Tongji University

Objectives of this lecture

This lecture will enable students to

- be familiar with the content of SA documents
- be able to write SA documents according to the template

SA definition

The software architecture of a system is the *set of structures* needed to reason about the system, which comprise software elements, relations among them, and properties of both.

What is a view? And a structure?

Modern software systems are complex.

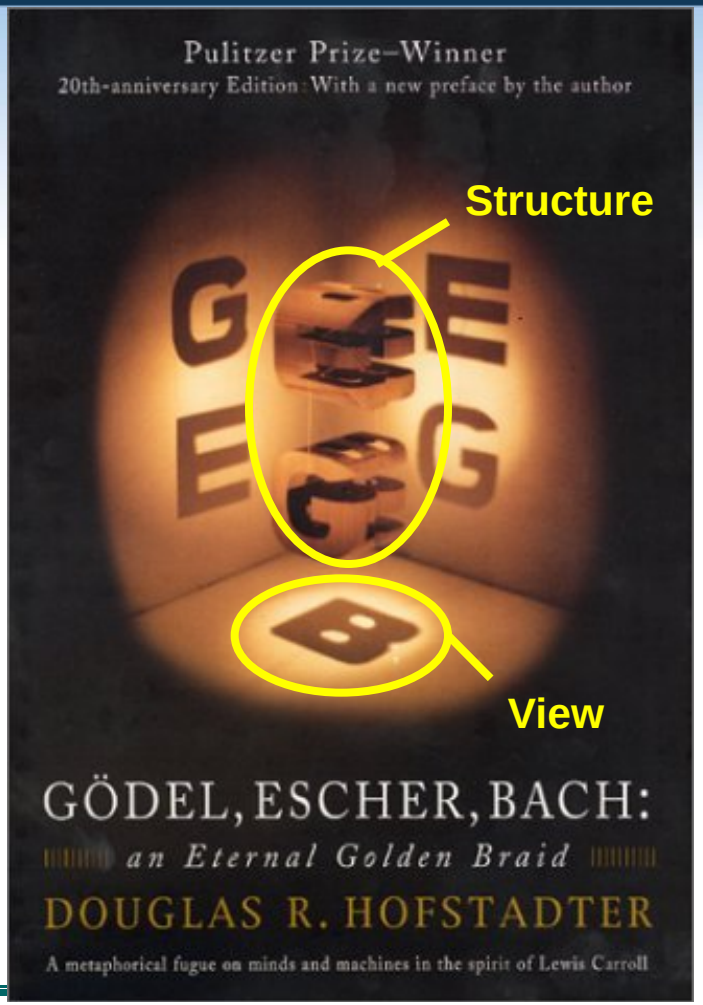
We focus at any time on a small number of issues of the system.

View

- Representation of a coherent set of architectural elements, as read/written by system stakeholders (plus relations)

Structure

- Set of architectural elements as they exist in software or hardware



What is a view? And a structure?

A view is a representation of a structure.

- A module structure is the set of the system's modules and their organization
- A module view is the representation of that structure, documented according to a template in a chosen notation, and used by some system stakeholders

Architects design structures. They document views of those structures

SA structures

Module structure

- Decomposition
- Uses
- Layered
- Class
- Data model

C&C structure

- Service
- Concurrency

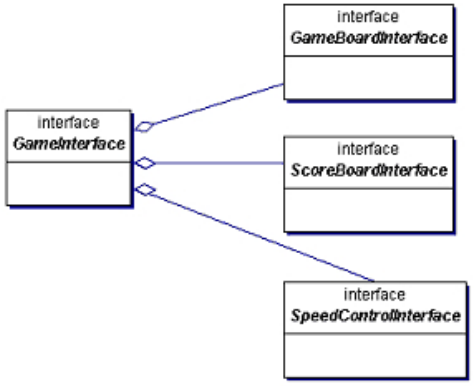
Allocation structure

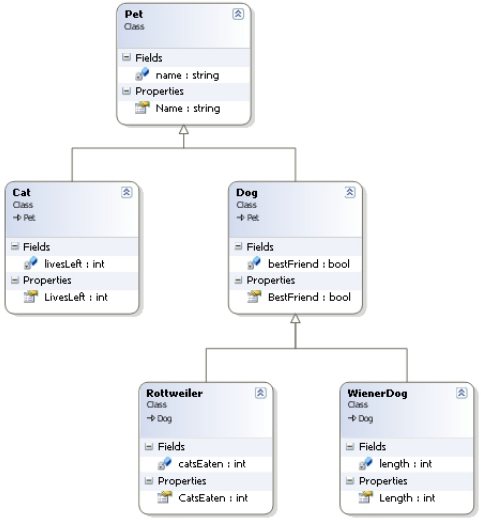
- Deployment
- Implementation
- Work assignment

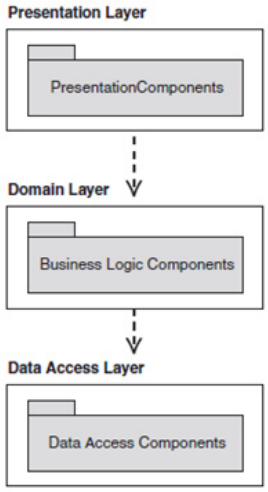
Module structures

Module structures

- The elements are modules such as classes, layers.
- Focus on the functional responsibility of each module, not emphasize runtime issue.



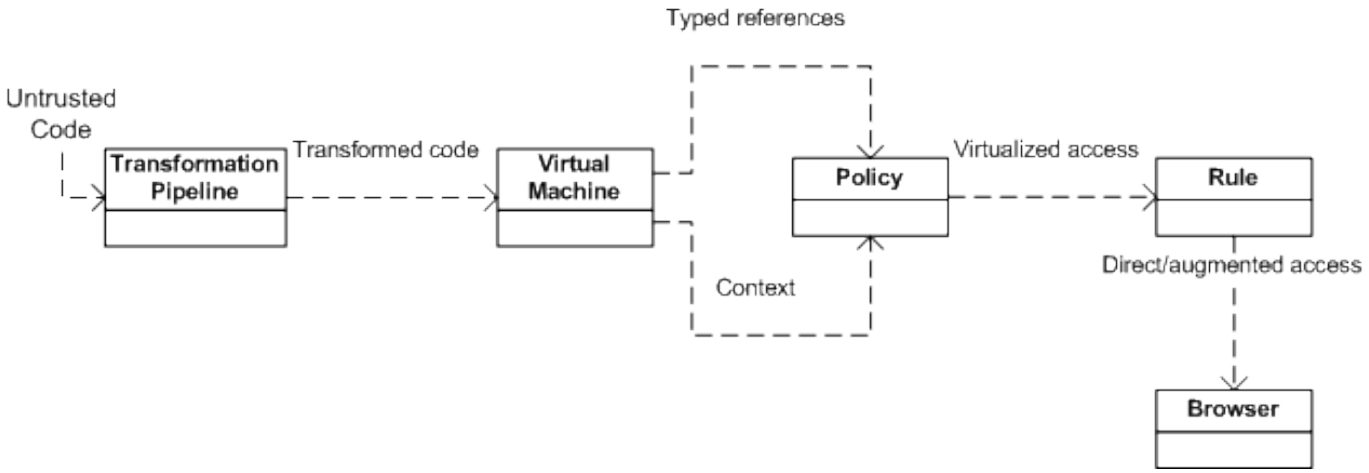




C&C structures

Component-and-connector structures

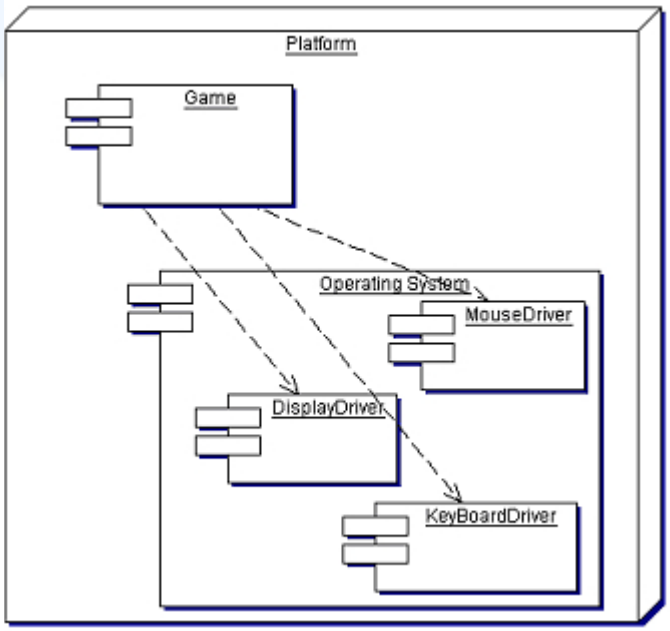
- The elements are runtime components (such as services, peers, clients, servers, filters) and connectors (such as call-return, process synchronization operators, pipes).

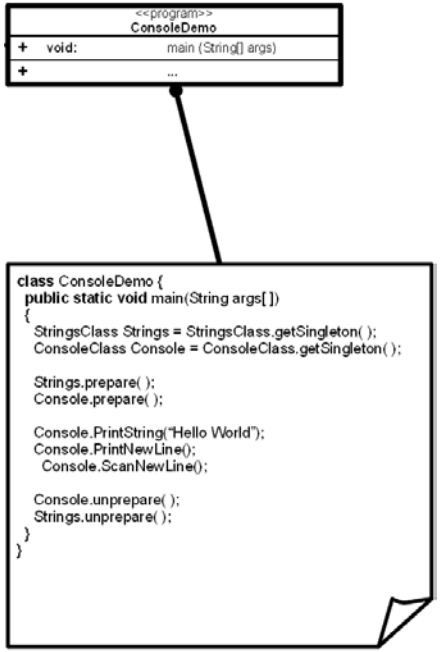


Allocation structures

Allocation structures

- They show the relationship between the software elements and elements in one or more external environments.





Relating structures to each other

Not all systems consider many structures

- Big vs small

Structures

- Main engineering leverage points of an architecture
- Bring with them the power to manipulate one/more quality attributes
- Powerful separation of concerns approach
- Useful for architecture documentation

Which structures to choose?

Many approaches:

1. Kruchten, 1995: Four+1
   - Focus on four structures
     - Logical, process, development, physical
   - Rational Unified Process
2. Soni, Nord, Hofmeister, 1995:
   - Conceptual, module, execution, code

Which structures to choose?

Among architect's obligations

- Understand how the various structures lead to quality attributes
- Choose the structures that will best deliver those attributes

Summary of architectural structures

We often think of system's structure in terms of its functionality

There are system properties in addition to functionality

- Physical distribution
- Process communication
- Synchronization, etc

Each structure: related to quality attributes

- *Uses structure*: engineered to build an extendable system
- *Process structure*: to eliminate deadlocks and bottlenecks
- *Decomposition structure*: to build a modifiable system

SA documentation

Significance

- Good architecture is useless if not understood or wrongly understood
- A basis for analysis
- Blueprint for construction

SA documentation

Prescriptive

- Prescribes what should be true by placing constraints on decisions to be made

Descriptive

- Describes what is true by recounting decisions already made about system design

Different stakeholders have different needs

- For information kinds, levels, treatments
- Stakeholder should quickly find the relevant documentation

SA documentation

Uses of SA documentation

- Key means to educate new people
- Primary vehicle for communication among stakeholders
  - *Same architect*: repository of thought, storehouse of design decisions
  - *Different architect*: check how predecessors tackled difficult tasks, why some decisions made
- Basis for system analysis and construction

Views and SA documentations

A SA is complex which needs multiple views to represent.

- A view contains a *particular* type of system elements.
  - e.g. a layered view show layers

Basic principle of documenting SA: Documenting the architecture is a matter of

1. Documenting the relevant views
2. Adding documentation that applies to more than one view

Views

Module views

C&C views

Allocation views

Views

Module views

C&C views

Allocation views

Module views

Categories:

- Decomposition views
- Uses views
- Layers views

Elements in module views:

- any decomposition unit: classes, layers…
  - Properties described: name, responsibility, visibility interfaces, implementation information, test information, inplementation constrains…

Relations in module views:

- is part of, depends on, is a

Module views

Usages:

- Blueprint for construction of the code
- Change-impact analysis
- Planning incremental development
- Requirements traceability analysis
- …

Views

Module views

C&C views

Allocation views

C&C views

Categories:

- Service-oriented architecture
- client-server views
- communication process views

Elements:

- Runtime entities – components
  - processes, objects, clients, servers, data stores…
- Pathways of interaction
  - communication links and protocol, information flows, access to shared storage…

C&C views

Categories:

- Service-oriented architecture
- client-server views
- communication process views

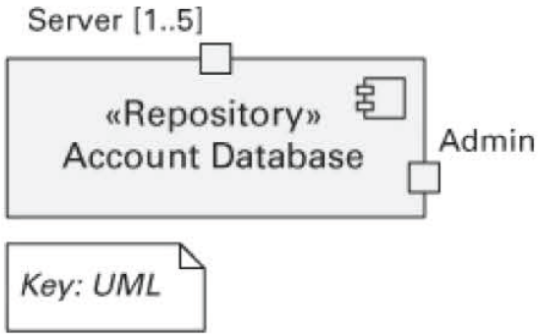
Elements:

- Connectors
  - service invocations, asynchronous message queues, pipes…

C&C views

Other related concepts:

- Port: interface of component
  - defines a point of potential interaction of a component with environment
  - can be replicated



- Roles: interface of connector
  - defines the ways in which the connector many be used by components to carry out interaction
  - e.g. Client-server connector might have *invokes-services* and *provides-services* roles; a pipe might have *writer* and *reader* roles.
  - can be replicated

C&C views

Associated properties of elements of a C&C view:

- Name
- Type
- Others depend on the type of component or connector
  - e.g. For performance analysis: latencies, queue capacities, thread priorities
  - Other properties: reliability, performance, functionality, security…

C&C view

Usages:

- Show how the system works.
- Guide development by specifying structure and behavior of runtime elements.
- Help reason about runtime system quality attributes.

Views

Module views

C&C views

Allocation views

Allocation views

Elements

- Software element
- Environmental element

Relations in allocation view

- allocated to
  - a mapping from software elements to environmental elements

Usages:

- For reasoning about performance, availability, security…
- For reasoning about distributed development and allocation of work to teams.
- …

Choosing relevant views

Needed documentation package is based on

- Who the stakeholders are
- The future uses of documentation
- Quality attributes
- Size of the system

At least one module view, one C&C view, and for larger systems, one allocation view

- In addition, there is a three-step method for choosing the views.

Choosing relevant views

Step 1. Produce candidate view list (matrix)

- List stakeholders
- List needed views
- Fill in cell with amount info: none, overview only, moderate detail, high detail

Choosing

Example of table

TABLE 9.2 Stakeholders and the Architecture Documentation They Might Find Most Useful

	Module Views				C&C Views	Allocation Views	
Stakeholder	**Decomposition**	**Uses**	**Class**	**Layer**	**Various**	**Deployment**	**Implementation**
Project Manager	s	s		s		d	
Member of Development Team	d	d	d	d	d	s	s
Testers and Integrators		d	d		s	s	s
Maintainers	d	d	d	d	d	s	s
Product Line Application Builder		d	s	o	s	s	s
Customer					s	o	
End User					s	s	
Analyst	d	d	s	d	s	d	
Infrastructure Support	s	s		s		s	d
New Stakeholder	x	x	x	x	x	x	x
Current and Future Architect	d	d	d	d	d	d	s

Key: d = detailed information, s = some details, o = overview information, x = anything

Source: Adapted from [Clements 03].

Choosing relevant views

Step 2. Combine views

- Too many views
- Remove views with “overview only” info or that serve very few stakeholders
- See if stakeholders of the above can be served by other views with more needed info
- Combine views

Choosing relevant views

Step 3. Prioritize

- Decide what to do first
  - Release (high-level) decomposition view early.
  - Don’t have to satisfy all the information.
  - Don’t have to complete one view before starting another.

Documenting a view

Principle of architecture documentation

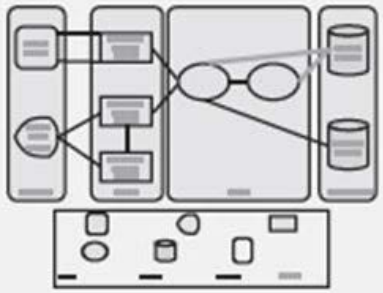
1. Documenting the relevant views
2. Adding documentation that applies to more than one view

Documenting a view

View Template

Template for a View

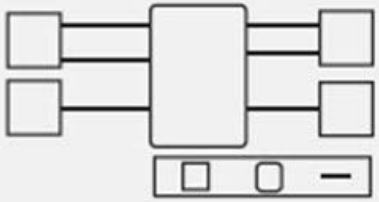
Section 1. Primary Presentation



Section 2. Element Catalog

- Section 2.A. Elements and Their Properties
- Section 2.B. Relations and Their Properties
- Section 2.C. Element Interfaces
- Section 2.D. Element Behavior

Section 3. Context Diagram



Section 4. Variability Guide

Section 5. Rationale

Section 1: Primary presentation

Elements that populate the view and relationships among them

- Should contain the information you wish to convey about the system - in the vocabulary of that view
- Not necessarily all of them
  - e.g. normal operation here, exception and error handling in other parts

Section 1: Primary presentation

Usually graphical, sometimes tabular

- If your primary presentation is graphical, make sure to include a key that explains the notation
- If that text is presented according to certain stylistic rules, these rules should be stated or incorporated by reference, as the analog to the graphical notation key.
- Present a terse summary of the most important information

Section 2: Element catalog

Details at least the elements and relationships shown in primary presentation

Backup for primary presentation

Elements and relations omitted from primary presentation

- Belong here
  - Introduced and explained

Describes

- The properties of elements
- The properties of relations
- The behavior of elements
- The interfaces of elements

Section 3: Context diagram

Shows how system in the view relates to environment in the vocabulary of view

Example: C&C view

- Show which component and connectors interact with external components and connectors
- Via which interfaces and protocols

Purpose: depict the scope of a view

Entities in the environment may be humans, other computer systems, or physical objects, such as sensors or controlled devices

Section 4: Variability guide

Shows how to exercise any variation points part of the architecture in the view

Example of variability: product lines

Includes documentation about each point of variation in architecture, including

- The options among which the choice is to be made
  - Module view: various parameterizations of modules
  - C&C view: constraints on replication, scheduling, protocol choice
  - Allocation view: conditions of allocation
- The binding time of the option
- Design, build, runtime

Section 5: Rationale

Explains why the design reflected in the view came to be.

The goal is to explain

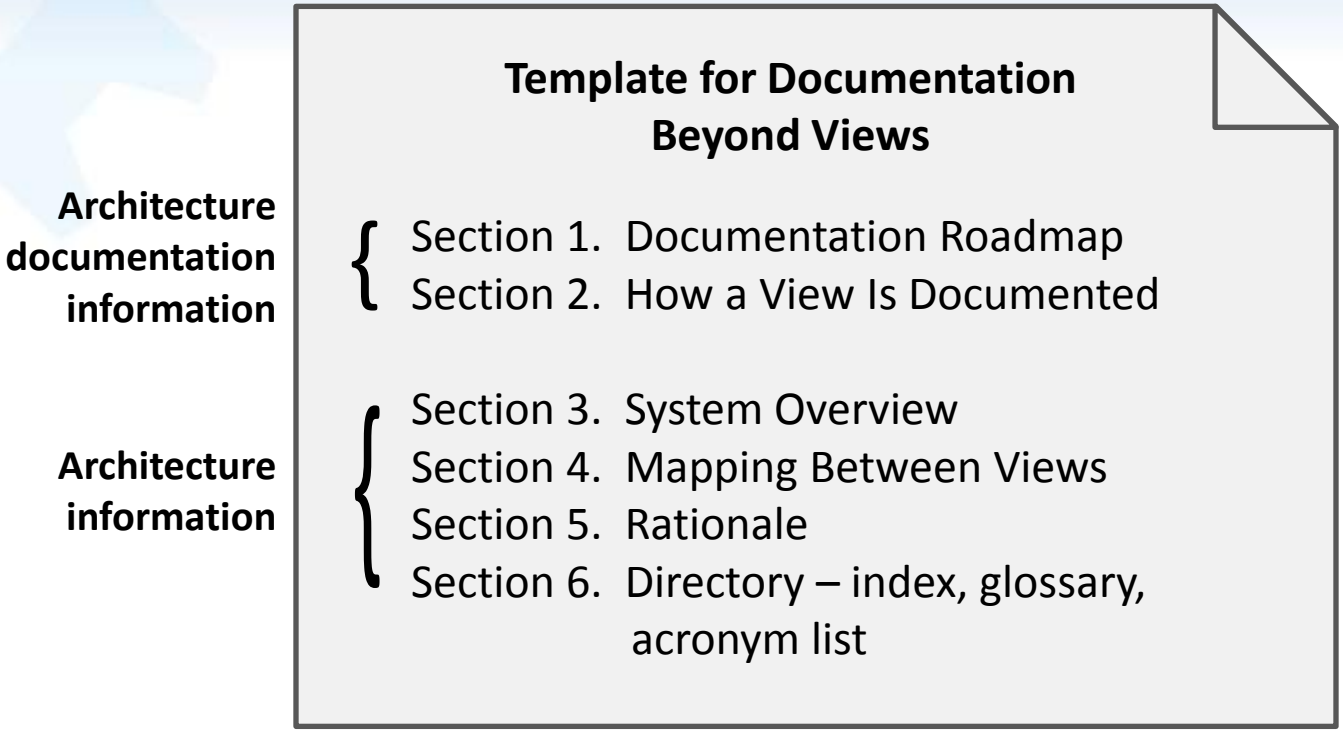
- why design is as it is
- provides convincing argument that it is sound

The choice of a pattern should be justified by

- describing the architectural problem and
- the rationale for choosing it.

Documenting information beyond views

Template for documentation beyond views



Section 1. Documentation roadmap

Tells reader what information is in the document and where to find it.

- Scope and summary
- How the document is organized
- View overview
  - the name of view and pattern
  - a description of element types, relation types, and property types
  - a description of language, modeling techniques…
- How stakeholders can use the documentation

Section 2. How a view is documented

Explain the standard organization used to document views.

- either the one described in the lecture or one of your own
- using a template

Section 3. System overview

A short prose description of system's function, users, background or constraints.

Section 4. Mapping between views

Help reader understand the associations between views.

- using tables

Section 5. Rationale

This section documents the architectural decisions that apply to more than one view.

- describe the decisions about which fundamental architecture patterns to use

Section 6. Directory

The directory is a set of reference material.

- index of terms, a glossary, and an acronym list

Documenting behavior

Structural information (views) not enough

- e.g. deadlock possibilities not captured

Behavior documented about

- how architecture elements interact with each other

Behavior descriptions add info on

- ordering of interactions among elements
- opportunities for concurrency
- time dependencies of interactions

Notations for documenting behavior

Trace-oriented language

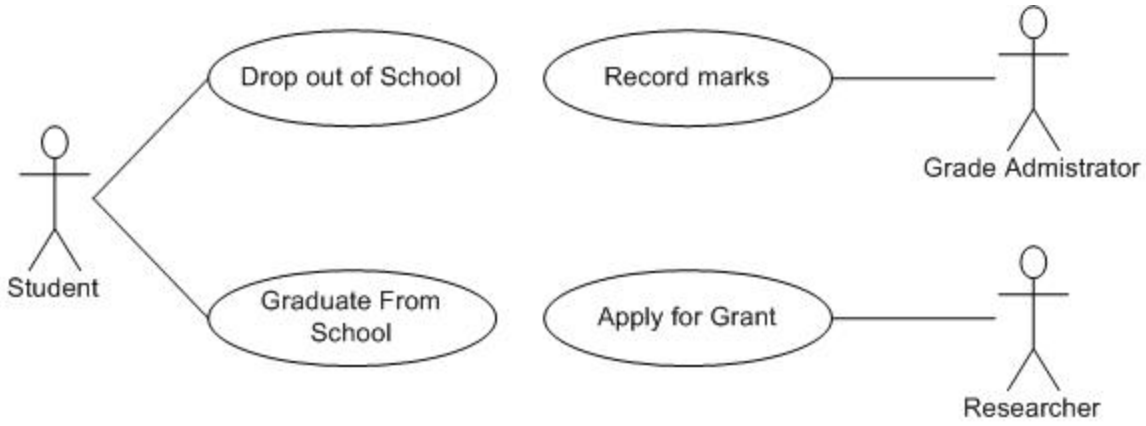
Comprehensive languages

Trace-oriented language

A trace describes a sequence of activities or interactions between elements.

Four notations for documenting traces:

- Use cases
- Sequence diagrams
- Communication diagrams
- Activity diagrams

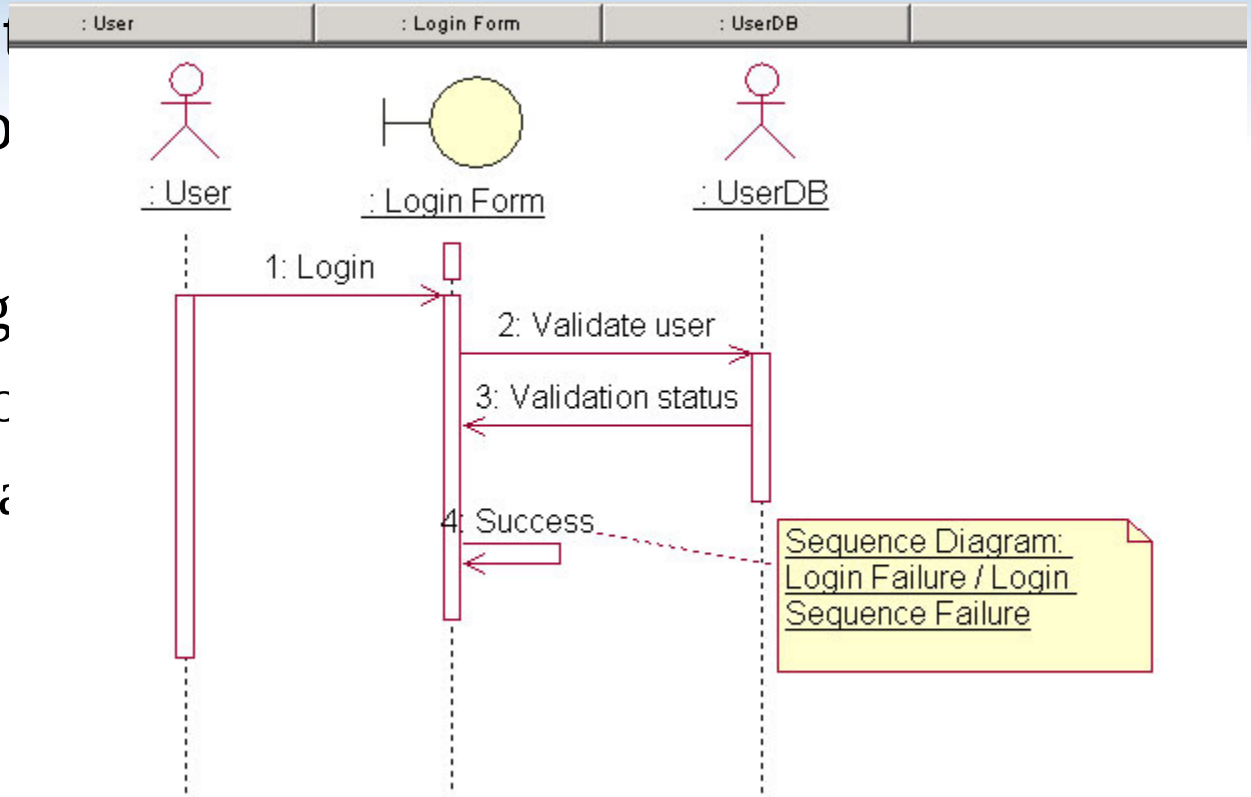


Trace-oriented language

A trace describes a sequence of activities or interactions between elemen

Four notations fo

- Use cases
- Sequence diag
- Communicatic
- Activity diagra

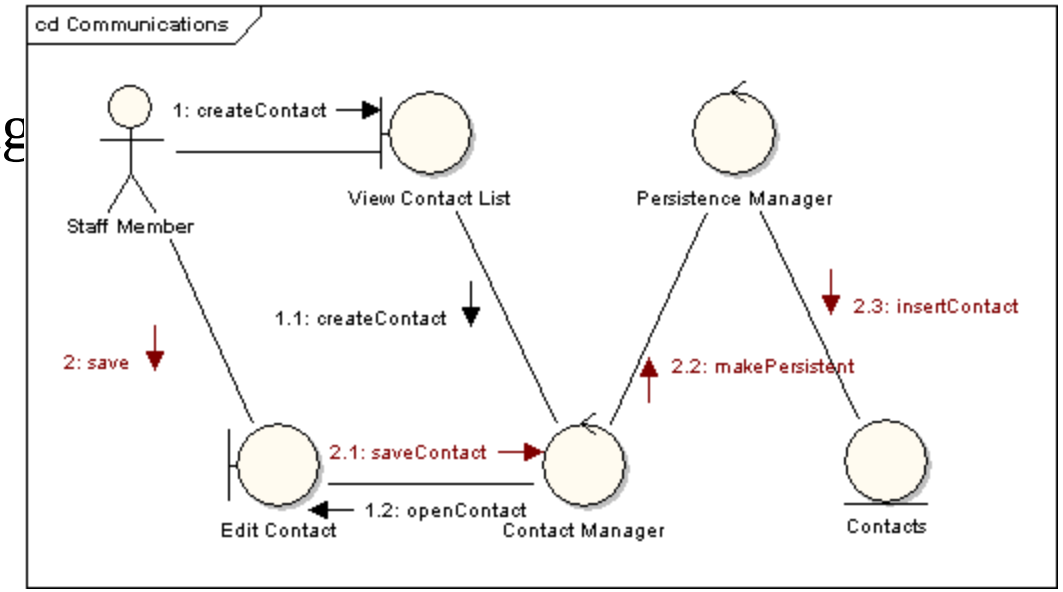


Trace-oriented language

A trace describes a sequence of activities or interactions between elements.

Four notations for documenting traces:

- Use cases
- Sequence diagrams
- Communication diag
- Activity diagrams

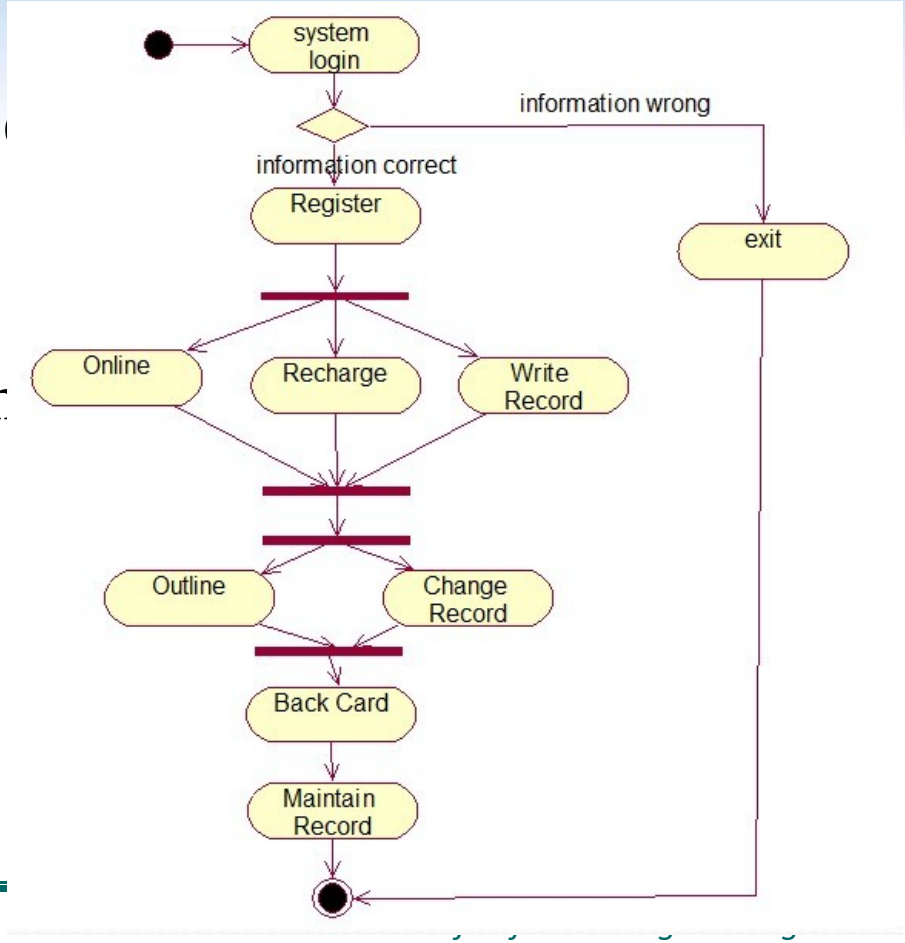


Trace-oriented language

A trace describes a sequence of activities or interactions between elements.

Four notations for docum

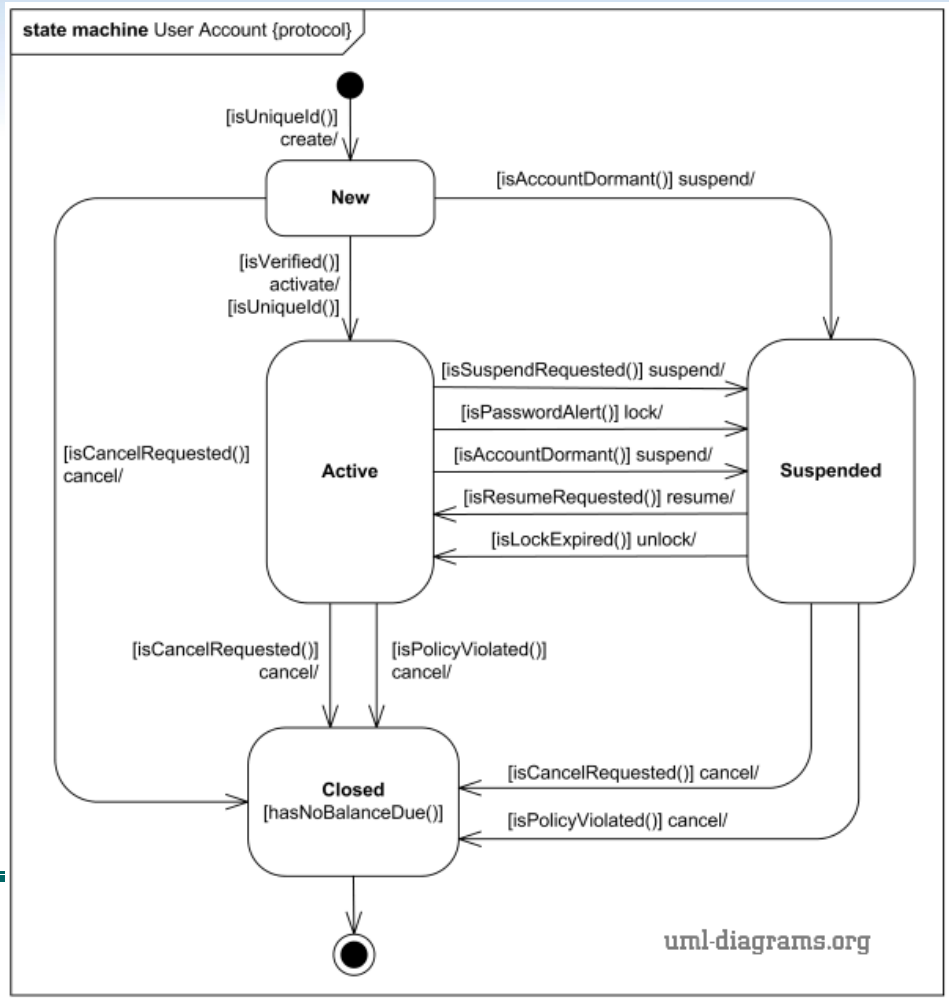
- Use cases
- Sequence diagrams
- Communication diagrar
- Activity diagrams



Comprehensive language

Comprehensive models show the complete behavior of structural elements.

- State machine diagram



Summary

SA views

- Module views
- C&C views
- Allocation views

SA documentation

- Software Architecture Views
- Architecture Documentation Beyond Views

The End